A Methodology for Deriving the Architectural Implications of Different Degrees of Mobility in Information Systems

Matthias Book, Volker Gruhn, Malte Hülder, Clemens Schäfer[1]
Chair of Applied Telematics / e-Business, University of Leipzig, Germany[2]

Abstract. When building information systems that can be accessed through desktop and mobile devices, developers often face the same basic design decisions that depend on a number of still unstructured criteria. Going through the whole decision-making process for every project is inefficient and error-prone, however, a comprehensive set of best practices has not yet been established. We therefore present the foundations of a classification scheme for mobile commerce systems that helps developers to categorize applications according to high-level requirements. After a discussion of the criteria, we suggest implications that can be drawn from it and present examples for their application.

Keywords. Mobile Computing, Software Architecture

1. Introduction

Recent years have brought an increasing demand of users to be able to do business very flexibly – ideally, any service should be available on any device, anywhere, anytime. The wide-spread presence of wireless networks and the availability of a diverse range of terminal devices has enabled the development of mobile applications that take us a step closer to accomplishing Weiser's vision of ubiquitous computing [1]. As this field is maturing, there is a need for collecting characteristics and principles of mobile systems. Considering that this is still a relatively young field, with most of the enabling technologies introduced into markets in the 1990s (such as GSM, GPRS and IEEE 802.11 WLAN) or just becoming available around this time (such as UMTS), the development methods for mobile applications are still unrefined and bear potential for optimization.

We are focusing on the architecture of mobile information systems with distinct client-server characteristics here since a large base of such information sys-

[1]Correspondence to: Clemens Schäfer, Chair of Applied Telematics / e-Business, University of Leipzig, Klostergasse 3, 04109 Leipzig, Germany. Tel.: +49-341-97-32334; Fax: +49-341-97-32339; E-mail: schaefer@ebus.informatik.uni-leipzig.de

[2]The Chair of Applied Telematics / e-Business is endowed by Deutsche Telekom AG.

tems, albeit without mobile capabilities, already exists today. Many organizations (e.g. in the insurance sector) are currently evaluating the business processes supported by these legacy applications for efficiency optimizations that may be gained through mobile access. Consequently, the architectural implications of enabling mobile access to information systems are an important issue in the industry.[1]

We consider it likely that some aspects of the development of mobile information systems are very similar to the development of traditional information systems, while other aspects may be more specific to mobile use, yet still independent of concrete, individual applications. Therefore, we hypothesize that a classification of information systems' mobile aspects will yield insights into their characteristics that support an efficient design process. Concrete measures to support architectural, technical, and process design decisions based on the proposed classification scheme are subject to further research in order to reach an ambitious goal: Using such a classification scheme, developers would not have to make fundamental design decisions on architectural, technical and process aspects from scratch during the development of each individual application, but could deduce some of the required design patterns, methods and processes from the class that an application belongs to.

While applications can certainly be classified quite exactly according to technical criteria, we believe that this should not be done *a priori* since the technical characteristics should be determined by the tasks the user needs to perform with the application, not vice versa. Also, the technical criteria are more prone to become outdated, while a classification on a higher level of abstraction remains applicable to a wide spectrum of technologies [2]. In fact, our motivation is to classify applications by non-technical criteria, as far as possible, so we can deduce the necessary technical implications from them.

In the following sections, we will give a brief overview of existing classification schemes that have been proposed (section 2). Next, we present our definitions of different aspects of mobility and connectivity, which are pivotal criteria of our classification methodology (section 3). After showing technical implications of our classification methodology (section 4) and depicting some examples of its use, we finally conclude with a discussion of the criteria introduced so far, and present opportunities for further research on additional criteria that can contribute to refining the classification scheme (section 5).

2. Related Work

Classification schemes for mobile computing applications that can be found in the literature are usually based on business criteria, i.e. they classify applications by the tasks and processes the user can perform with them. For example, Varshney and Vetter [3] discern mobile or wireless financial, advertising, inventory

[1]Obviously, other classes of mobile computing applications without explicit client-server characteristics also exist (e.g. peer-to-peer applications based on mobile ad-hoc networks). However, for the sake of clarity of the classification scheme, we restrict ourselves to mobile information systems for now. An extension of the classification scheme to include other types of applications will be an important aspect of further research (see section 5).

management, product locating and shopping, service management, auctioning, entertainment and office applications, among others.

However, the criteria for associating specific applications with these classes are not explicitly stated, and the classes themselves do not seem to be disjoint, which renders the association ambiguous and thus makes it hard to draw precise conclusions from the classification. Varshney and Vetter do present a list of networking requirements that need to be fulfilled for applications of certain classes (such as location management, multicast support, network dependability, quality of service, and roaming across networks). However, since quite a few of these requirements are listed for every class, they do not really seem to be correlated to the class characteristics, but are actually basic requirements for any mobile application to a stronger or lesser degree.

Classification approaches that employ technology-centric criteria, such as the underlying network protocols, may seem helpful at first glance since the selected technology can have far-reaching consequences for the application design. For example, the decision to use a web-based user interface based on HTTP implies that the browser must use the pull paradigm to communicate with the server, while the server cannot initiate any communication with the client. This imposes restrictions on the interaction patterns that users can perform within the application [4]. For another example, developers of GPRS-based applications need to bear in mind latencies of about 25 seconds for setting up a connection and about two seconds for a request-response cycle in addition to the actual data transmission time [5]. For an application that relies on many request-response cycles (e.g. piecing a web page together from various elements), this technical characteristic from a low level of the protocol stack may severely diminish usability.

From yet another perspective, we can distinguish different types of mobility and use them for the classification of applications. In this context, Pandya [6] discerns device mobility, user mobility and service mobility: According to his definition, device mobility is given when the device remains connected to a network while it is moved physically. User mobility means that the user is not restricted to a single device, but can rather use different devices in order to use a service. Finally, a user experiences service mobility when he can access the same service from anywhere, independently of his location.

This approach of abstracting from concrete business and technology characteristics seems to be a step in the right direction. However, we believe that the concept of device mobility needs to be refined to encompass different degrees of mobility, as described in section 3. Pandya's user mobility, in contrast, should always be given in today's mobile applications: A device that is tied so inseparably to one user that it cannot be used by a different user, and that the user cannot use a different device to accomplish the same task, is virtually inconceivable. Thus, user mobility can hardly serve as a distinguishing criterion for mobile applications. Finally, Pandya's definition of service mobility seems like a combination of device and user mobility, which makes it equivalent to device mobility if we assume user mobility to be always given. Specifically, service mobility does not seem to imply that the software providing the service is mobile.

In their roadmap, Roman, Picco and Murphy [7] make the distinction between physical mobility (movement of mobile hosts) and logical mobility (mobile units of code and state). From our point of view the situation here is similar to Pandya's definition: In our opinion the notion of physical mobility of Roman et al. is too coarse and needs to be refined. The same is true for the notion of mobility used by Issarny et al. [8]. Here the notion of computer and user mobility is used, but these notions lack a proper definition and hence a solid foundation.

Due to these concerns with existing classification approaches, we want to sharpen the existing definitions and thus present an alternative methodology that abstracts from concrete business and technology characteristics, but instead is based on high-level usage patterns, device capabilities, and service requirements.

3. Classification Criteria

In the following sections, we will present the three classification criteria user mobility, device mobility and service connectivity, and discuss their correlation.

Since today's e-commerce applications usually provide a range of different services to users (where a *service* shall be defined as a part of an application that supports a certain business process), it may be difficult to associate the whole application unambiguously with one category. Therefore, our criteria allow the independent classification of individual services within an application.

3.1. User Mobility

Since our goal is to deduce architectural, infrastructural and implementation aspects of a service from its classification, we strive to keep the classification criteria as non-technical as possible, and instead focus on user requirements and usage patterns. Consequently, the first criterion expresses the level of freedom of location and movement that is granted to a user while he is executing a business process (the user's location and movement while he is *not* executing a process are irrelevant, since the system is neither aware of nor influenced by them). We define four degrees of *user mobility*:

- A *local user* can only execute business processes at the application's location.
- A *distributed user* can execute business processes from a remote location.
- A *mobile user* can execute business processes from different remote locations.
- An *in-motion user* can execute business processes while changing his remote location.

Looking at the sets of all local, distributed, mobile and in-motion users (denoted by U_{loc}, U_{dis}, U_{mob} and U_{mot}, respectively), we notice that the sets of local and distributed users are obviously disjoint ($U_{\text{loc}} \cap U_{\text{dis}} = \emptyset$). Mobile users, however, are special cases of distributed users, and in-motion users are mobile users with additional requirements. Thus, distributed users are a superset of mobile users, which in turn are a superset of in-motion users ($U_{\text{dis}} \supset U_{\text{mob}} \supset U_{\text{mot}}$). The distinction between the latter user groups becomes clear if we consider the differences between the sets, i.e. distributed users that are not mobile ($U_{\text{dis}} \setminus U_{\text{mob}}$), and mobile, yet not in-motion users ($U_{\text{mob}} \setminus U_{\text{mot}}$):

- A distributed *immobile user* can execute business processes from one remote location only.
- A mobile *at-rest user* can execute business processes while remaining in a static remote location only.

Note that in order to execute a business process while in motion, the user must carry the device that is used to work with the service with him (e.g. a PDA). However, if the user will aways be at rest while executing the process, he does not need to carry his own device with him, but can use any devices that are provided in locations where users may intend to work with the application (e.g. terminals dispersed throughout a trade fair center).

After defining the user's mobility, we now focus on the mobility of the device that he uses to execute business processes.

3.2. Device Mobility

Intuitive definitions of object mobility usually focus on whether it is possible to move an object physically, e.g. by defining mobile as "capable of moving or of being moved readily from place to place" [9]. From a software engineering perspective, though, this definition of physical mobility is not sufficient because it does not state how to distinguish the "places" from each other and thus deduce that the object has moved – we still need a frame of reference in relation to which objects are moving.

Communications networks provide such a frame of reference in the form of their access points (e.g. GSM cells, WLAN hot spots, etc.). Every access point has a certain coverage area that a device must enter in order to be able to connect to the network. In this scenario, we can define that *mobility is the capability of moving or of being moved readily between the coverage areas of a network's access points.* According to this definition, we distinguish four degrees of *device mobility*:

- A *local* device cannot connect to the network.
- A *distributed* device can connect to the network.
- A *mobile device* can connect to different access points.
- An *in-motion device* can connect to different access points while the user is using the device.

The sets of all local, distributed, mobile and in-motion devices (denoted by D_{loc}, D_{dis}, D_{mob} and D_{mot}, respectively) exhibit the same relationships as the different degrees of user mobility: $D_{\text{loc}} \cap D_{\text{dis}} = \emptyset$, and $D_{\text{dis}} \supset D_{\text{mob}} \supset D_{\text{mot}}$. The distinction between the network-enabled devices again becomes clearer when we look at the differences $D_{\text{dis}} \setminus D_{\text{mob}}$ and $D_{\text{mob}} \setminus D_{\text{mot}}$ that define immobile and at-rest devices:

- A distributed *immobile device* can always connect to the same access point only.
- A mobile *at-rest device* can only connect to the same access point while the user is using the device.

Note that in the presence of multiple distinct communication networks with different access point densities, a device that is mobile in relation to one network (i.e. moving into and out of the coverage areas of different access points) may at the same time be immobile in relation to another network (i.e. remaining within the coverage area of the same access point at all times). It is therefore important to consider which network to use as a frame of reference.

Since virtually every object can be moved from place to place with sufficient effort, the word "readily" introduces an important constraint both into the general and the specific definition of mobility presented above: While not specifying absolutely under which conditions an object can be considered mobile, it demands that the effort required to move an object shall be evaluated in relation to other metrics (e.g. the benefit of moving it or the effort required to move other objects). In the general definition, this effort may be determined by the object's weight or fixation. In the network-specific definition, configuring a device appropriately to connect to different access points may require additional effort, which may render the device immobile in the sense that it cannot connect *readily* to different access points.

We express devices' mobility in terms of their capability to connect readily to different access points here, because from a software engineering standpoint, small physical movements within the coverage area of one access point are undetectable and thus irrelevant.[2] However, larger physical movements that take the device out of reach of one access point and into the coverage area of another require a handover process that must either be handled transparently by the network infrastructure or explicitly by the application. Consequently, device mobility is not a characteristic of the device alone, but also of the network infrastructure, and potentially of the application.

3.3. Service Connectivity

Following the OSI reference model [10], the mobility of the service is determined by the mobility of the device that allows the remote user to execute business processes. Therefore, service mobility is not an independent criterion, but tied to device mobility. However, services must be prepared to handle a side effect of device mobility: Since the coverage areas of a network's access points may not overlap everywhere, there may be locations without network coverage. Consequently, a mobile device may be unable to connect to the network in certain locations, and an in-motion device may experience a loss of connection when passing through an uncovered area. The provisions that need to be taken by the service in order to handle these situations depend on how strongly it relies on the network connection. We define four degrees of *service connectivity*:

- An *offline service* never requires a network connection.
- A *hybrid-offline service* occasionally requires a network connection.
- A *hybrid-online service* requires a network connection most of the time.
- An *online service* requires a network connection at all times.

[2]Note that we are only concerned with the impact of mobility on applications' architecture and communications infrastructure here. For location-aware applications, a small physical movement may certainly have consequences; however, those are in the realm of the business logic. An examination of the ties between device mobility and location-awareness is a topic of ongoing research, as discussed in section 5.

	D_{loc}	$D_{\text{dis}} \setminus D_{\text{mob}}$	$D_{\text{mob}} \setminus D_{\text{mot}}$	D_{mot}
U_{loc}	S_{off}	S_{off}	S_{off}	S_{off}
$U_{\text{dis}} \setminus U_{\text{mob}}$	$\emptyset$	$S_{\text{hoff}} \cup S_{\text{hon}} \cup S_{\text{on}}$	$S_{\text{hoff}} \cup S_{\text{hon}} \cup S_{\text{on}}$	$S_{\text{hoff}} \cup S_{\text{hon}} \cup S_{\text{on}}$
$U_{\text{mob}} \setminus U_{\text{mot}}$	$\emptyset$	S_{hoff}	$S_{\text{hoff}} \cup S_{\text{hon}} \cup S_{\text{on}}$	$S_{\text{hoff}} \cup S_{\text{hon}} \cup S_{\text{on}}$
U_{mot}	$\emptyset$	S_{hoff}	$S_{\text{hoff}} \cup S_{\text{hon}}$	$S_{\text{hoff}} \cup S_{\text{hon}} \cup S_{\text{on}}$

Table 1. Correlations between the classification criteria.

The sets of all offline, hybrid-offline, hybrid-online and online services (denoted by S_{off}, S_{hoff}, S_{hon} and S_{on}, respectively) are disjoint ($S_{\text{off}} \cap S_{\text{hoff}} \cap S_{\text{hon}} \cap S_{\text{on}} = \emptyset$).

One might argue that the definitions of the hybrid connectivity degrees are somewhat fuzzy, which is certainly true. Similarly to the "readiness" for movement in the previous section, the limiting ratio of connected vs. disconnected operations can hardly be specified absolutely here. Rather, the tolerable level of disconnected operation must be set in relation to other metrics such as the degree of autonomy that is allowed by the service's business process, and any given architecture or infrastructure elements.

Usually, a hybrid-offline service will allow the user to execute a business process offline, and only require a connection to the server infrequently and briefly in order to transmit the process' input and/or output data. In contrast, a hybrid-online service will usually communicate frequently with the server while the user is executing a business process, but handle temporary losses of connection gracefully (e.g. by caching input and/or output data that is transmitted as soon as a connection is available, or performing trivial tasks autonomously without requiring a connection).

3.4. Correlations Between the Criteria

Considering possible combinations of the three criteria user mobility, device mobility and service connectivity in an application, we find that some combinations are feasible while others are contradictory.

For example, an in-motion user aims to work while he is changing his location. If he is using a device from the mobile at-rest category (i.e. a device that can connect to the network from any location, but not while moving), he will not be able to use an online service, because that service category requires a permanent network connection, which a mobile at-rest device cannot provide while the user is in motion. However, an in-motion user *can* employ a mobile at-rest device if the service is hybrid-online or hybrid-offline, i.e. the service does not require a network connection at all times but also allows some level of disconnected operation.

Relationships like this one are summarized in Table 1: If a user belongs to a certain group from the leftmost column and uses a device belonging to a certain group in the topmost row, then the feasible service connectivity must be from the set given in the associated table cell. For a user from the U_{mot} group and a device from the $D_{\text{mob}} \setminus D_{\text{mot}}$ group, for example, we find that the set of feasible service connectivity degrees is $S_{\text{hoff}} \cup S_{\text{hon}}$.

Further to this discussion, if an in-motion user (U_{mot}) employs a distributed immobile device ($D_{\text{dis}} \setminus D_{\text{mob}}$), i.e. one that can only connect to the network from a single, specific location), the service can only be hybrid-offline (S_{hoff}), i.e. not require a network connection most of the time (while the user is moving), and only briefly connect when the user is at rest in the specific location that the device can connect from. Conversely, if an in-motion user employs an in-motion device that can remain connected to the network even when the user is moving (D_{mot}), he can use online services that require a permanent connection, or hybrid-online services if he wants to be immune against occasional connection losses. He may even use a hybrid-offline service, even though it would not utilize the device's networking capabilities fully.

A closer look at the table suggests that with the exception of U_{loc}, D_{loc} and S_{off}, the correlation between the three criteria follows these rules:

- If a certain service connectivity is feasible for a certain user/device mobility combination, all lower service connectivities are also feasible for this combination.
- In order for online service connectivity to be feasible, the device must be in the same or a higher mobility category than the user.
- If the device is in a lower mobility category than the user, the feasible service connectivity is reduced, possibly overproportionally.

Local users, local devices and offline services do not follow these rules because their sets are disjoint from the network-capable degrees of the three criteria. If a local user only works with a service on the same device that it is running on, no network connection is necessary, so the service is classified as offline (even if the device is capable of accessing a network, that feature will not be utilized by the service). Conversely, if the user is not local, no offline service can be available on any device since a network connection is always required to contact it.

One might argue that a moving user could still use an offline service (such as a premium calculator) on a mobile device that he is carrying with him. However, since we defined user mobility as movement occuring remotely from a service and device mobility as movement occuring in relation to a network, the user, device and service in this scenario would all be local and stationary in relation to each other. From a technical perspective, the physical mobility of this closed system is not relevant. Therefore, it also belongs to the local user and device category. Note, however, that these offline services may still support the user in executing business processes on hybrid or online services.

4. Technical Implications of the Classification

So far, our methodology enables developers to deduce feasible degrees of service connectivity from the frame conditions set by the desired user mobility and the available device mobility.[3] Often, these different degrees of connectivity will

[3]Depending on the given information, the correlation table can obviously also be used to deduce the degree of user mobility allowed by a given combination of device mobility and service connectivity, or to indicate the level of device mobility required to provide a given service to users with a given mobility degree.

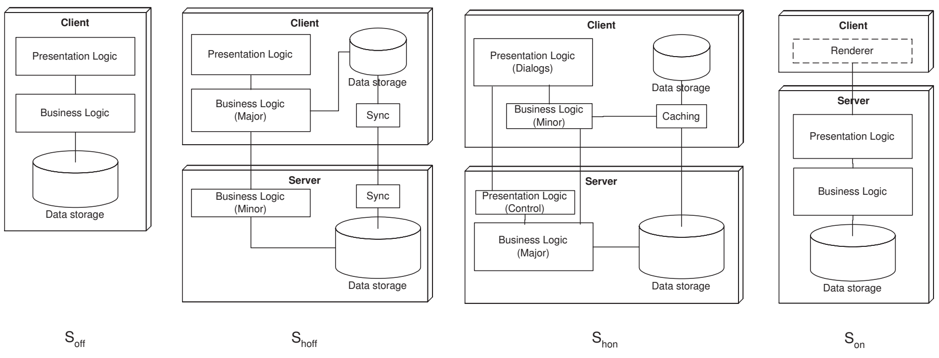

Figure 1. Distribution patterns for information systems, depending on service connectivity.

require different technical provisions to be taken in the implementation of the service.

In the second step of our methodology, we therefore need to identify architectural implications of different degrees of connectivity. We are focusing on information systems with client-server characteristics here – other types of mobile computing applications (e.g. telemetry or peer-to-peer applications) [3] may be subject to different architectural implications, which are a topic of future research.

4.1. Distribution of Architectural Tiers

E-commerce systems store and process data according to the business requirements of their application domain. Their architecture usually follows a three-tier model, distinguishing presentation logic, business logic and data storage. In a client-server environment, these layers may be distributed according to different strategies. Figure 1 shows how these distribution patterns depend on the degree of service connectivity. For each set of services, the figure indicates which parts of the three layers must reside on the client, which parts must reside on the server, and what information must be communicated between both sides.

For example, a hybrid-offline service may access the network occasionally, but is used without a connection most of the time. Therefore, it must also implement the complete presentation logic and all major features of the business logic on the client side, and store all data that the user requires to execute business processes. In most information systems, this is a subset of the complete data repository stored on the server, which is synchronized when a connection is present. In addition, the server may provide some minor additional business logic features when connected.

Conversely, a hybrid-online service can rely on the presence of a network connection most of the time, and may just have to deal with temporary losses of connection. Under these conditions, most of the dialog control logic can reside on the server, however, the client should be able to display dialogs and allow basic interactions even without a network connection. This can be achieved by implementing the dialogs on the client and coupling them loosely to the server-side control logic that just needs to send “triggers” to let the client perform predefined dialog sequences autonomously. Similarly, most of the business logic can be implemented on the server, with just a restricted set of features being available on the client in case of a temporary loss of connection. All required data can usually be accessed over the network connection, however, a temporary storage is required on the client to cache incoming data and buffer outgoing data in case the connection breaks down. Once the connection has been re-established, the cached/buffered data can be updated from/in the server-side database.

Finally, an online service can rely on a permanent network connection, so the presentation logic, business logic and data storage can reside completely on the server. Communication is then constantly required in order to transmit a description of the user interface to display to the client. Since the interpretation and rendering of the interface description can be realized application-independently (e.g. by an existing web browser), we used a dashed box in the figure.

Note that the figure shows the *minimum client requirements* and the *maximum server requirements* for providing the associated service connectivity. For example, if more presentation logic was implemented on the client side of an online service, the connectivity degree could remain the same (in fact, it would be equivalent to a hybrid-online service for the presentation layer). However, if no data was stored on the client side of a hybrid-online service, online connectivity would be required for the storage layer. These relationships can be expressed in the following rules:

- Different layers of one service can support different connectivity degrees.
- Shifting logic from the server to the client on a layer allows a lower connectivity degree for that layer
- Shifting logic from the client to the server on a layer requires a higher connectivity degree for that layer
- A service's overall connectivity degree is determined by the highest degree of all its layers.

For example, a service that runs the complete presentation logic on the client, relies on the server for most business logic operations, and reads and writes any processed data directly from/to a database on the server can only support online connectivity.

4.2. Examples

To illustrate the classification methodology, we will now apply it to real-world examples from recent industry projects. After identifying the requirements and deducing the available architecture choices as suggested in this paper, we compare them to the architectures that were actually implemented in those projects.

As an example, we consider a dispatch system for a truckage company that constantly provides truck drivers (who are in motion most of the time and thus belong to the U_{mot} category) with current dispatch and routing information and allows them to trigger processes in the back-end through an on-board unit connected to the GSM network (e.g. an in-motion device from the D_{mot} category). This combination would allow the whole range of hybrid-offline, hybrid-online and online service connectivity. However, since parts of the system (such as the routing) still need to work while crossing gaps in the network coverage, an architecture that does not exclusively rely on the network was chosen: While the complete presentation logic and business logic required by the driver resides on the client (as for a hybrid-offline service), the system strives to always work with current data received from the server, and only uses cached data in case the connection is interrupted. Since this represents a hybrid-online connectivity in the storage layer, the whole system is considered hybrid-online (S_{hon}).

In contrast, a lottery portal that allows users to participate in lottery games with their WAP-enabled cellphone also fulfills the U_{mot} (in-motion users) and D_{mot} (in-motion devices) criteria. However, to keep the service as independent from the users' devices as possible, the system was built using a completely server-centered architecture. Only the markup for the displayed WML pages is communicated to the client, resulting in online service connectivity (S_{on}).

The same portal can also be accessed on another channel via short message service (SMS). This way, an in-motion user (U_{mot}) may type his lottery bet into a text message on his mobile phone (a D_{mot} device). The message may be composed independently of network availability and is sent out when network coverage is available. The server then generates a confirmation SMS, which is transmitted back to the device when it is connected to the network. The system is therefore still usable (i.e. messages can be composed and read) during brief network outages. However, if the network connection is lost for a longer time, it becomes more unusable since the user cannot place bets or receive the important confirmation messages. Since there is no business logic residing on the client, and client-side data (i.e. message) storage represents a kind of buffering, the service can be considered hybrid-online (S_{hon}).

5. Conclusions

In the preceding sections, we introduced three criteria for the classification of mobile information systems that allow us to deduce the possible connectivity of a service from two high-level frame conditions – coarse usage patterns (in terms of user mobility) and basic device capabilities (in terms of device mobility). Using the service connectivity for guidance, we can then decide on the more technical issue of how to distribute the implementation of the presentation, business and storage layer across the client-server architecture.

Due to the lack of space we could not show in detail how the second step may be performed, which is also still being optimizied as part of our ongoing research. In this context, we are mainly focusing on two questions: Firstly, which other application classes need to be considered? Can we use any existing application classification and apply our new criteria to it in order to examine the implications of mobile use, or are there other general characteristics of mobile applications that determine the distribution patterns of other information systems? And secondly, when a certain combination of user and device mobility allows a range of service connectivity degrees, which criteria determine which connectivity degree is actually chosen on each layer?

When looking for such classification criteria, we need to find a balance between mobile application characteristics whose nature is more that of requirements than of consequences or features (for example, high bandwidth is a feature of certain communication channels – a related requirement that could be more suitable as a classification criterion is high data volume). Also, while as software engineers, we tend to focus on technical characteristics, we should also look for relevant application domain criteria – one strong candidate for such criteria certainly is location awareness.

Further research should also go into the granularity that is possible and sensible for such a classification scheme. The criteria presented in this paper already operate on the service instead of the application level, so different services can constitute one application; and within one service, different layers can (under certain restrictions) support different service connectivities. However, we still need to examine architectural implications for applications that comprise services with different connectivity degrees.

Finally, we have only considered the mobility of users and devices with regard to physical networks so far. Ultimately, the classification criteria and rules for their architectural implications should also cover virtual aspects of mobility – i.e., mobility with regard to virtual networks, and logical mobility of code that may be exchanged between devices.

Based on the basic criteria introduced in this paper, the above questions should provide a basis for interesting research in a number of directions, helping us to understand the characteristics of mobile commerce applications better and thus develop them more efficiently.

References

[1] M. Weiser. The computer for the twenty-first century. *Scientific American*, 265(3):94–104, 1991.

[2] O. H, D. Spath, A. Weisbecker, A. C. Allag, U. Rosenthal, and M. Veit. Ein Klassifikationsschema für die Architektur von mobilen Anwendungen - Erläutert an einem Praxisbeispiel zur mobilen Erfassung von Führerscheinprüfungen. In *Mobile Business – Processes, Platforms, Payments*, pages 131–142. Gesellschaft für Informatik, 2005.

[3] Upkar Varshney and Ron Vetter. Mobile commerce: framework, applications and networking support. *Mobile Networks and Applications*, 7(3):185–198, 2002.

[4] J. Rice, A. Farquhar, P. Piernot, and T. Gruber. Using the web instead of a window system. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI '96)*, 1996.

[5] Rajiv Chakravorty and Ian Pratt. WWW performance over GPRS. In *Proceedings of the IEEE International Conference on Mobile and Wireless Communication Networks*, 2002.

[6] R. Pandya. *Mobile and personal communication systems and services.* IEEE Press, 2000.

[7] G.-C. Roman, G. P. Picco, and A. L. Murphy. Software Engineering for Mobility: A Roadmap. In *Proceedings of the Conference on the Future of Software Engineering*, pages 241–258. ACM Press, 2000.

[8] V. Issarny, F. Tartanoglu, Jinshan Liu, and F. Sailhan. Software Architecture for Mobile Distributed Computing. In *Proceedings of the Fourth Working IEEE/IFIP Conference on Software Architecture (WICSA'04)*, pages 201–210. IEEE, 2004.

[9] AHD. *The American Heritage Dictionary of the English Language*. Houghton Mifflin Co, 4th edition, 2000.

[10] ISO. *ISO/IEC Standard 7498-1: Information Processing Systems – OSI Reference Model – The Basic Model*. International Organization for Standardization, 1994.